538'S STAR WARS SURVEY

Getting started

Load the data. You'll need to put the data file in your current working directory, or equivalent to this load() function, you can double-click it. Use the str() function to take a peek at the types of variables in the data.

The survey (and it's results) are described here:
https://fivethirtyeight.com/features/americas-favorite-star-wars-movies-and-least-favorite-characters/

```
> load('StarWarsSurvey.RData')
> str(starwars)
```

```
'data.frame':	1186 obs. of  38 variables:
 $ RespondentID                 : num  3.29e+09 3.29e+09 3.29e+09 3.29e+09 3.29e+09 ...
 $ SeenAnyMovie                 : Factor w/ 3 levels "No","Yes",NA: 2 1 2 2 2 2 2 2 2 2 ...
 $ AreYouAFan                   : Factor w/ 3 levels "No","Yes",NA: 2 3 1 2 2 2 2 2 2 1 ...
 $ SeenEpisodeI                 : num  1 0 1 1 1 1 1 1 1 0 ...
 $ SeenEpisodeII                : num  1 0 1 1 1 1 1 1 1 1 ...
 $ SeenEpisodeIII               : num  1 0 1 1 1 1 1 1 1 0 ...
 $ SeenEpisodeIV                : num  1 0 0 1 1 1 1 1 1 0 ...
 $ SeenEpisodeV                 : num  1 0 0 1 1 1 1 1 1 0 ...
 $ SeenEpisodeVI                : num  1 0 0 1 1 1 1 1 1 0 ...
 $ RankEpisodeI                 : num  3 0 1 5 5 1 6 4 5 1 ...
 $ RankEpisodeII                : num  2 0 2 6 4 4 5 5 4 2 ...
 $ RankEpisodeIII               : num  1 0 3 1 6 3 4 6 6 3 ...
 $ RankEpisodeIV                : num  4 0 4 2 2 6 3 3 2 4 ...
 $ RankEpisodeV                 : num  5 0 5 4 1 5 1 2 1 5 ...
 $ RankEpisodeVI                : num  6 0 6 3 3 2 2 1 3 6 ...
 $ ViewHanSolo                  : num  5 0 4 5 5 5 5 5 5 3 ...
 $ ViewLukeSkywalker            : num  5 0 4 5 4 5 5 4 2 5 ...
 $ ViewPrincessLeiaOrgana       : num  5 0 4 5 4 5 4 5 4 5 ...
 $ ViewAnakinSkywalker          : num  5 0 4 5 2 5 4 3 4 5 ...
 $ ViewObiWanKenobi             : num  5 0 4 5 5 5 5 5 4 5 ...
 $ ViewEmperorPalpatine         : num  5 0 0 4 1 3 5 1 5 2 ...
 $ ViewDarthVader               : num  5 0 0 5 4 5 5 2 5 5 ...
 $ ViewLandoCalrissian          : num  0 0 0 4 3 3 5 3 5 2 ...
 $ ViewBobaFett                 : num  0 0 0 2 5 4 5 4 5 2 ...
 $ ViewC3P0                     : num  5 0 0 5 4 4 4 4 3 5 ...
 $ ViewR2D2                     : num  5 0 0 5 4 4 5 4 4 5 ...
 $ ViewJarJarBinks              : num  5 0 0 5 1 4 2 1 1 5 ...
 $ ViewPadmeAmidala             : num  5 0 0 5 4 3 4 2 2 2 ...
 $ ViewYoda                     : num  5 0 0 5 4 5 5 5 4 5 ...
 $ WhichCharacterShotFirst      : Factor w/ 4 levels "Greedo","Han",..: 3 4 3 3 1 2 2 2 2 3 ...
 $ FamiliarWithExpandedUniverse: Factor w/ 3 levels "No","Yes",NA: 2 3 1 1 2 2 2 1 1 1 ...
 $ FanOfExpandedUniverse        : Factor w/ 3 levels "No","Yes",NA: 1 3 3 3 1 1 1 3 3 3 ...
 $ FanOfStarTrek                : Factor w/ 3 levels "No","Yes",NA: 1 2 1 2 1 2 1 2 1 1 ...
 $ Gender                       : Factor w/ 3 levels "Female","Male",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ Age                          : Factor w/ 5 levels "> 60","18-29",..: 2 2 2 2 2 2 2 2 2 2 ...
```

```
$ HouseholdIncome             : Factor w/ 6 levels "$0 - $24,999",..: 6 1 1 2 2 4 6 6 1 4 ...
$ Education                   : Factor w/ 6 levels "Bachelor degree",..: 3 1 3 5 5 1 3 3 5 5 ...
$ Location                    : Factor w/ 10 levels "East North Central",..: 7 9 8 8 8 3 1 7 7 6 ...
```

There are lots of factor variables which are not suited for PCA input (PCA functions will undoubtedly want you to input numeric data, regardless of software platform). While it's not the *most* principled thing to use dummy variables as input to PCA (we prefer to think about PCA in terms of elliptically distributed numeric data), it's common and often works. The math holds just the same - we still get a projection of our data that is of maximal variance. We're going to proceed with that approach here - though you are welcome to further exploration without the binary and factor columns.

To create the dummy variables for all factors, we make a model matrix with **one-hot encoded** factors (this means we don't leave out a reference column). The contrasts.arg= option in the following model.matrix() function will do the one-hot encoding.

```
> starwars.x = model.matrix(RespondentID~. ,
+             contrasts.arg = lapply(starwars[,sapply(starwars,is.factor) ],
+                                    contrasts, contrasts=FALSE),
+             data = starwars)
```

Print the dimensions of the final matrix (with dummy variables enumerated):

```
> dim(starwars.x)
```

```
[1] 1186   76
```

This is the *true dimensionality* of your data! If you were asked "how many columns are in your dataset?" you would be wise to answer 76 rather than 38!

Computing the PCA

The prcomp() function is the one I most often recommend for reasonably sized principal component calculations in R. This function returns a list with class "prcomp" containing the following components (from help prcomp):

1. sdev: the standard deviations of the principal components (i.e., the square roots of the eigenvalues of the covariance/correlation matrix, though the calculation is actually done with the singular values of the data matrix).

2. rotation: the matrix of *variable loadings* (i.e., a matrix whose columns contain the eigenvectors). The function princomp returns this in the element loadings.

3. x: if retx is true *the value of the rotated data (i.e. the scores)* (the centred (and scaled if requested) data multiplied by the rotation matrix) is returned. Hence, cov(x) is the diagonal matrix $diag(sdev^2)$. For the formula method, napredict() is applied to handle the treatment of values omitted by the na.action.

4. center, scale: the centering and scaling used, or FALSE.

The option scale = TRUE inside the prcomp() function instructs the program to use **correlation PCA**. The **default is covariance PCA**.

```
> pca = prcomp(starwars.x, scale =T)
```

An error message! Cannot rescale a constant/zero column to unit variance. Solution: check for columns with zero variance and remove them. Recheck dimensions of the matrix to see how many columns we lost.

```
> starwars.x = starwars.x[,apply(starwars.x, 2, sd)>0 ]
> dim(starwars.x)
```

```
[1] 1186   74
```

We can now compute the principal components of the matrix.

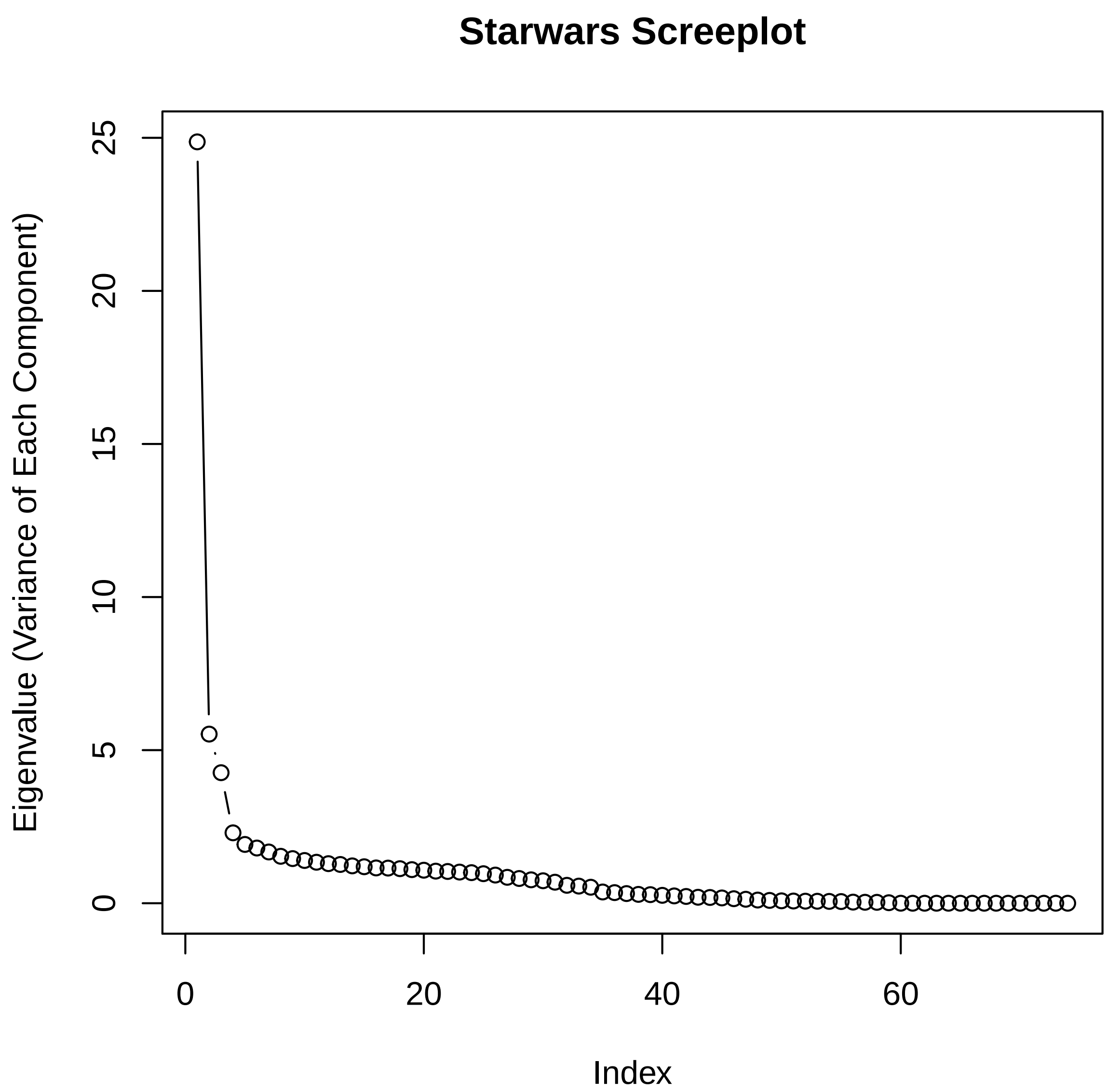
```
> pca = prcomp(starwars.x, scale =T)
> summary(pca)
```

```
Importance of components:
                          PC1     PC2     PC3     PC4
Standard deviation     4.9867 2.35026 2.06469 1.51646
Proportion of Variance 0.3361 0.07465 0.05761 0.03108
Cumulative Proportion  0.3361 0.41069 0.46830 0.49938
                           PC5     PC6     PC7     PC8
Standard deviation     1.38567 1.34283 1.29383 1.23810
Proportion of Variance 0.02595 0.02437 0.02262 0.02071
Cumulative Proportion  0.52532 0.54969 0.57231 0.59303
                          PC9    PC10    PC11    PC12
Standard deviation     1.2073 1.18157 1.15696 1.13694
Proportion of Variance 0.0197 0.01887 0.01809 0.01747
Cumulative Proportion  0.6127 0.63159 0.64968 0.66715
                          PC13    PC14    PC15    PC16
Standard deviation     1.12519 1.10475 1.09130 1.07371
Proportion of Variance 0.01711 0.01649 0.01609 0.01558
Cumulative Proportion  0.68425 0.70075 0.71684 0.73242
                         PC17    PC18    PC19    PC20
Standard deviation     1.0711 1.06231 1.04745 1.03829
Proportion of Variance 0.0155 0.01525 0.01483 0.01457
Cumulative Proportion  0.7479 0.76317 0.77800 0.79257
                          PC21    PC22    PC23    PC24
Standard deviation     1.02360 1.01837 1.00848 0.99930
Proportion of Variance 0.01416 0.01401 0.01374 0.01349
Cumulative Proportion  0.80673 0.82074 0.83448 0.84798
```

```
                          PC25    PC26    PC27    PC28
Standard deviation     0.98187 0.95920 0.92009 0.89868
Proportion of Variance 0.01303 0.01243 0.01144 0.01091
Cumulative Proportion  0.86101 0.87344 0.88488 0.89579
                          PC29    PC30    PC31    PC32
Standard deviation     0.87419 0.85653 0.82842 0.76485
Proportion of Variance 0.01033 0.00991 0.00927 0.00791
Cumulative Proportion  0.90612 0.91604 0.92531 0.93321
                          PC33    PC34    PC35    PC36
Standard deviation     0.74762 0.72291 0.60635 0.58850
Proportion of Variance 0.00755 0.00706 0.00497 0.00468
Cumulative Proportion  0.94077 0.94783 0.95280 0.95748
                          PC37    PC38    PC39    PC40
Standard deviation     0.56159 0.53827 0.52990 0.50785
Proportion of Variance 0.00426 0.00392 0.00379 0.00349
Cumulative Proportion  0.96174 0.96566 0.96945 0.97294
                          PC41    PC42    PC43    PC44
Standard deviation     0.48907 0.47064 0.44171 0.43370
Proportion of Variance 0.00323 0.00299 0.00264 0.00254
Cumulative Proportion  0.97617 0.97916 0.98180 0.98434
                          PC45   PC46    PC47   PC48
Standard deviation     0.41602 0.3846 0.35956 0.3224
Proportion of Variance 0.00234 0.0020 0.00175 0.0014
Cumulative Proportion  0.98668 0.9887 0.99042 0.9918
                          PC49    PC50    PC51    PC52
Standard deviation     0.30378 0.28043 0.26828 0.25947
Proportion of Variance 0.00125 0.00106 0.00097 0.00091
Cumulative Proportion  0.99308 0.99414 0.99511 0.99602
                          PC53    PC54    PC55   PC56
Standard deviation     0.25411 0.24497 0.23357 0.1920
Proportion of Variance 0.00087 0.00081 0.00074 0.0005
Cumulative Proportion  0.99689 0.99770 0.99844 0.9989
                          PC57   PC58    PC59    PC60
Standard deviation     0.17751 0.1712 0.13025 0.02501
Proportion of Variance 0.00043 0.0004 0.00023 0.00001
Cumulative Proportion  0.99937 0.9998 0.99999 1.00000
                           PC61      PC62      PC63
Standard deviation     2.34e-14 8.381e-15 6.354e-15
Proportion of Variance 0.00e+00 0.000e+00 0.000e+00
Cumulative Proportion  1.00e+00 1.000e+00 1.000e+00
                            PC64      PC65      PC66
Standard deviation     5.135e-15 3.778e-15 3.227e-15
Proportion of Variance 0.000e+00 0.000e+00 0.000e+00
Cumulative Proportion  1.000e+00 1.000e+00 1.000e+00
                            PC67      PC68      PC69
Standard deviation     2.875e-15 2.659e-15 2.074e-15
Proportion of Variance 0.000e+00 0.000e+00 0.000e+00
Cumulative Proportion  1.000e+00 1.000e+00 1.000e+00
                            PC70      PC71      PC72
Standard deviation     1.911e-15 1.411e-15 4.951e-16
Proportion of Variance 0.000e+00 0.000e+00 0.000e+00
Cumulative Proportion  1.000e+00 1.000e+00 1.000e+00
                            PC73      PC74
Standard deviation     3.793e-16 3.793e-16
Proportion of Variance 0.000e+00 0.000e+00
Cumulative Proportion  1.000e+00 1.000e+00
```

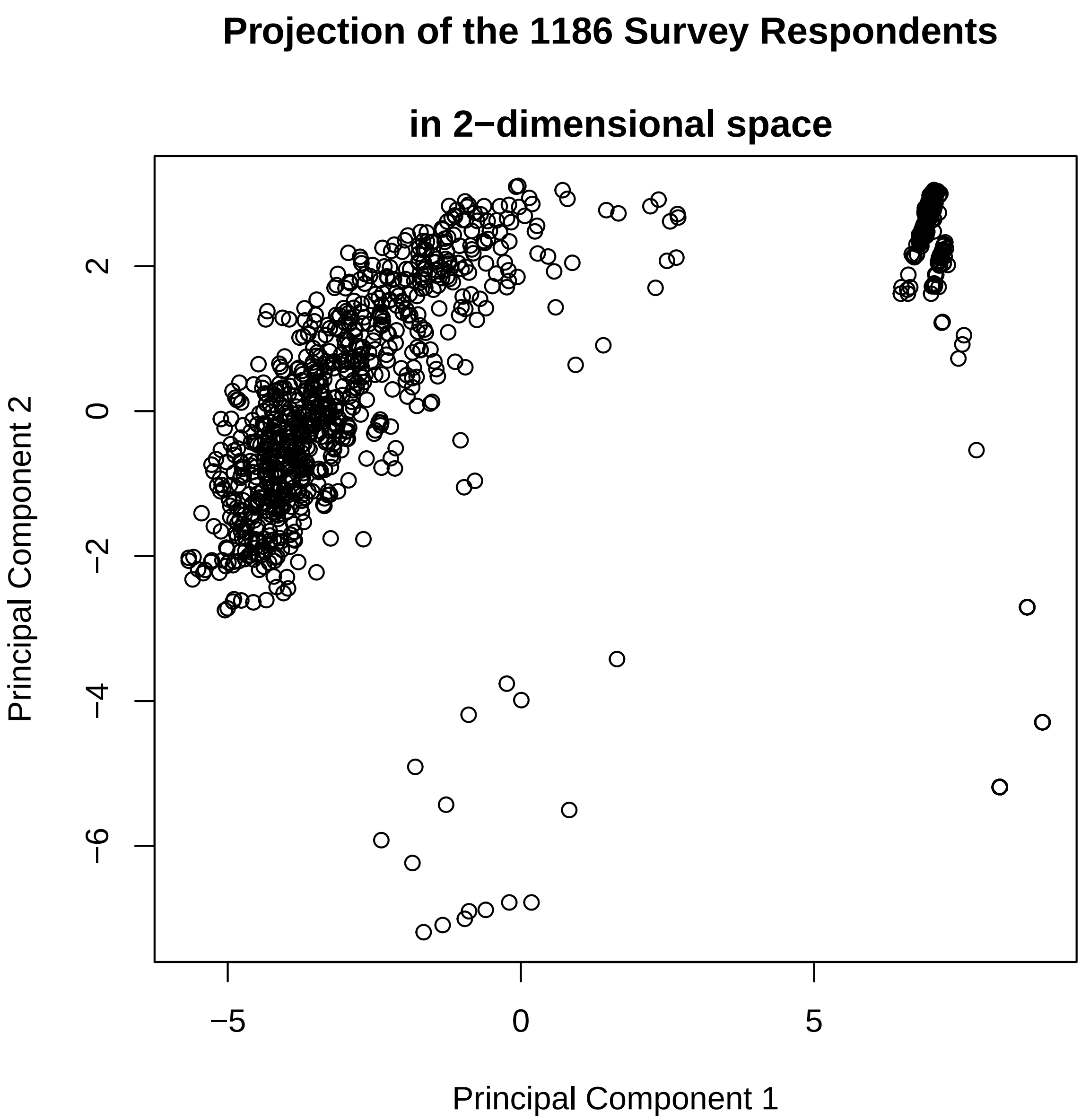
To get the screeplot:

```
> plot(pca$sdev^2, type='b', ylab = 'Eigenvalue (Variance of Each Component)',
+      main = 'Starwars Screeplot')
```


Starwars Screeplot
Eigenvalue (Variance of Each Component)
0
5
10
15
20
25
0
20
40
60
Index

Projection of data in 2 dimensions

```
> plot(pca$x[,1:2], xlab = "Principal Component 1",
+      ylab = "Principal Component 2",
+      main='Projection of the 1186 Survey Respondents
+         \n in 2-dimensional space')
```



It is common for points to have the same or very similar scores along PCs, particularly when each variable has a limited range of values (i.e. binary or likert scale). This means that many points can be overlapping on a plot and that can be misleading. A common solution is to "jitter" the plot which merely adds some random error to the placement of the points so that you can see overlapping groups of points with more clarity:

```
> plot(jitter(pca$x[,1:2],amount=0.1), xlab = "Principal Component 1",
+      ylab = "Principal Component 2",
+      main='Projection of the 1186 Survey Respondents
+         \n in 2-dimensional space (w/Jitter)')
```

Note the clusters of individuals outstanding on PC1. Let's investigate the dominant variables driving PC1. We'll simply look at the loadings of variables on PC1, ordered by the absolute value of their magnitude.

```
> head(pca$rotation[order(abs(pca$rotation[,1]),decreasing=T),1],10)
```

```
                  AreYouAFanNA
                     0.1936650
     WhichCharacterShotFirstNA
                     0.1920700
FamiliarWithExpandedUniverseNA
                     0.1920700
                   ViewHanSolo
                    -0.1908215
              ViewLukeSkywalker
                    -0.1904492
        ViewPrincessLeiaOrgana
                    -0.1901918
                      ViewR2D2
                    -0.1889610
                      ViewYoda
                    -0.1888073
               ViewObiWanKenobi
                    -0.1873507
                      ViewC3P0
                    -0.1840787
```

The 3 largest loadings involve levels from missing values. In this case, I'd like to just take a quick look at the data of interest, those observations that have scores on PC1 that are greater than 5. So I'll create a temporary dataset to explore

```
> look = starwars[pca$x[,1]>5, ]
> head(look)
```

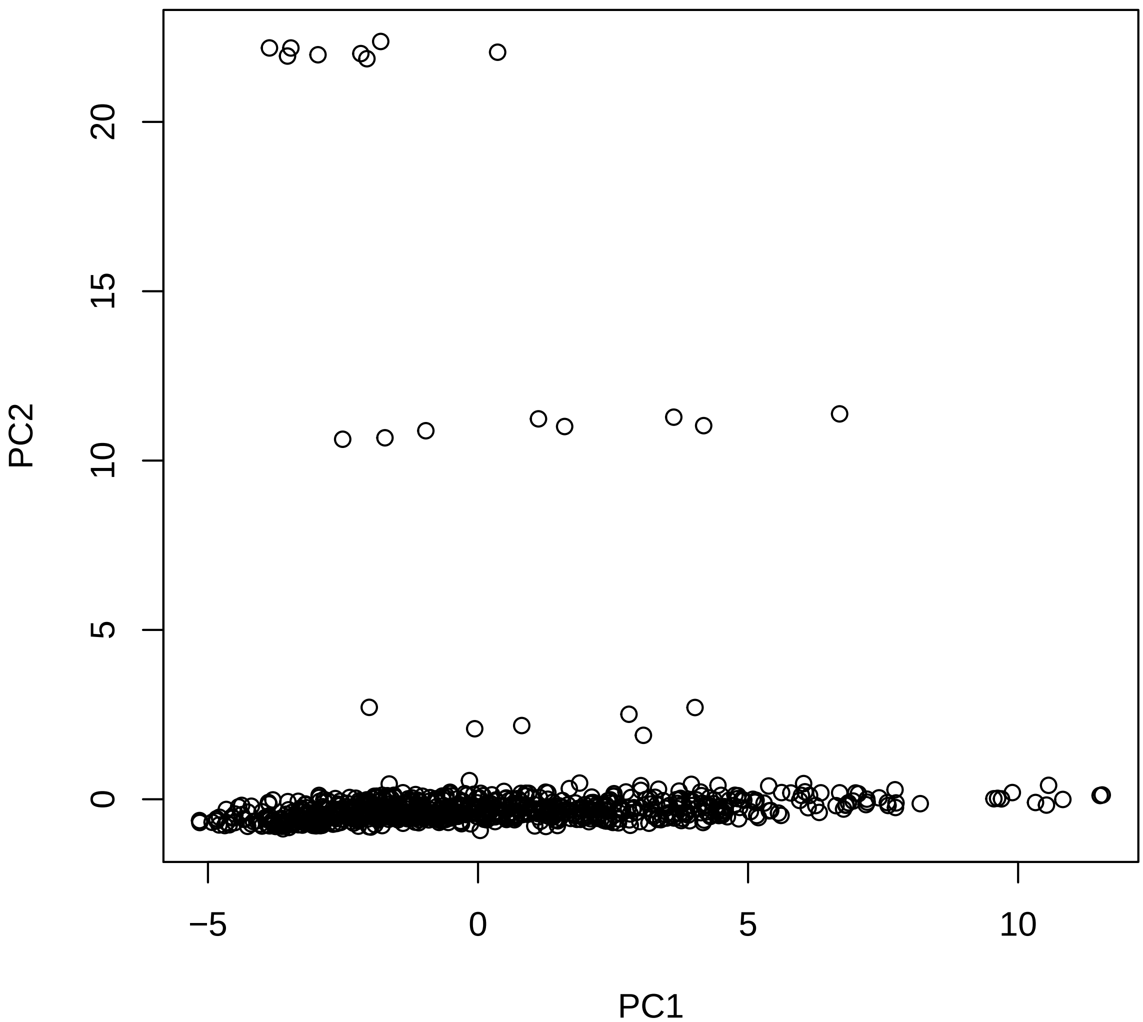
```
   RespondentID SeenAnyMovie AreYouAFan SeenEpisodeI
2    3292879538           No       <NA>            0
11   3292637870          Yes       <NA>            0
12   3292635062           No       <NA>            0
26   3292447974           No       <NA>            0
35   3292297848           No       <NA>            0
47   3292202220           No       <NA>            0
   SeenEpisodeII SeenEpisodeIII SeenEpisodeIV SeenEpisodeV
2              0              0             0            0
11             0              0             0            0
12             0              0             0            0
26             0              0             0            0
35             0              0             0            0
47             0              0             0            0
   SeenEpisodeVI RankEpisodeI RankEpisodeII RankEpisodeIII
2              0            0             0              0
11             0            0             0              0
12             0            0             0              0
26             0            0             0              0
35             0            0             0              0
47             0            0             0              0
   RankEpisodeIV RankEpisodeV RankEpisodeVI ViewHanSolo
2              0            0             0           0
11             0            0             0           0
12             0            0             0           0
26             0            0             0           0
35             0            0             0           0
47             0            0             0           0
   ViewLukeSkywalker ViewPrincessLeiaOrgana
2                  0                      0
11                 0                      0
12                 0                      0
26                 0                      0
35                 0                      0
47                 0                      0
   ViewAnakinSkywalker ViewObiWanKenobi
```

```
2                  0               0
11                 0               0
12                 0               0
26                 0               0
35                 0               0
47                 0               0
   ViewEmperorPalpatine ViewDarthVader ViewLandoCalrissian
2                     0              0                   0
11                    0              0                   0
12                    0              0                   0
26                    0              0                   0
35                    0              0                   0
47                    0              0                   0
   ViewBobaFett ViewC3P0 ViewR2D2 ViewJarJarBinks
2             0        0        0               0
11            0        0        0               0
12            0        0        0               0
26            0        0        0               0
35            0        0        0               0
47            0        0        0               0
   ViewPadmeAmidala ViewYoda WhichCharacterShotFirst
2                 0        0                    <NA>
11                0        0                    <NA>
12                0        0                    <NA>
26                0        0                    <NA>
35                0        0                    <NA>
47                0        0                    <NA>
   FamiliarWithExpandedUniverse FanOfExpandedUniverse
2                          <NA>                  <NA>
11                         <NA>                  <NA>
12                         <NA>                  <NA>
26                         <NA>                  <NA>
35                         <NA>                  <NA>
47                         <NA>                  <NA>
   FanOfStarTrek Gender   Age   HouseholdIncome
2            Yes   Male 18-29      $0 - $24,999
11          <NA>   <NA>  <NA>              <NA>
12          <NA>   <NA>  <NA>              <NA>
26           Yes   Male 18-29      $0 - $24,999
35           Yes   Male 30-44 $50,000 - $99,999
47            No   Male 18-29      $0 - $24,999
            Education           Location
2     Bachelor degree West South Central
11               <NA>               <NA>
12               <NA>               <NA>
26 High school degree East South Central
35    Graduate degree East South Central
47 High school degree            Pacific
```

```
> #View(look)
```

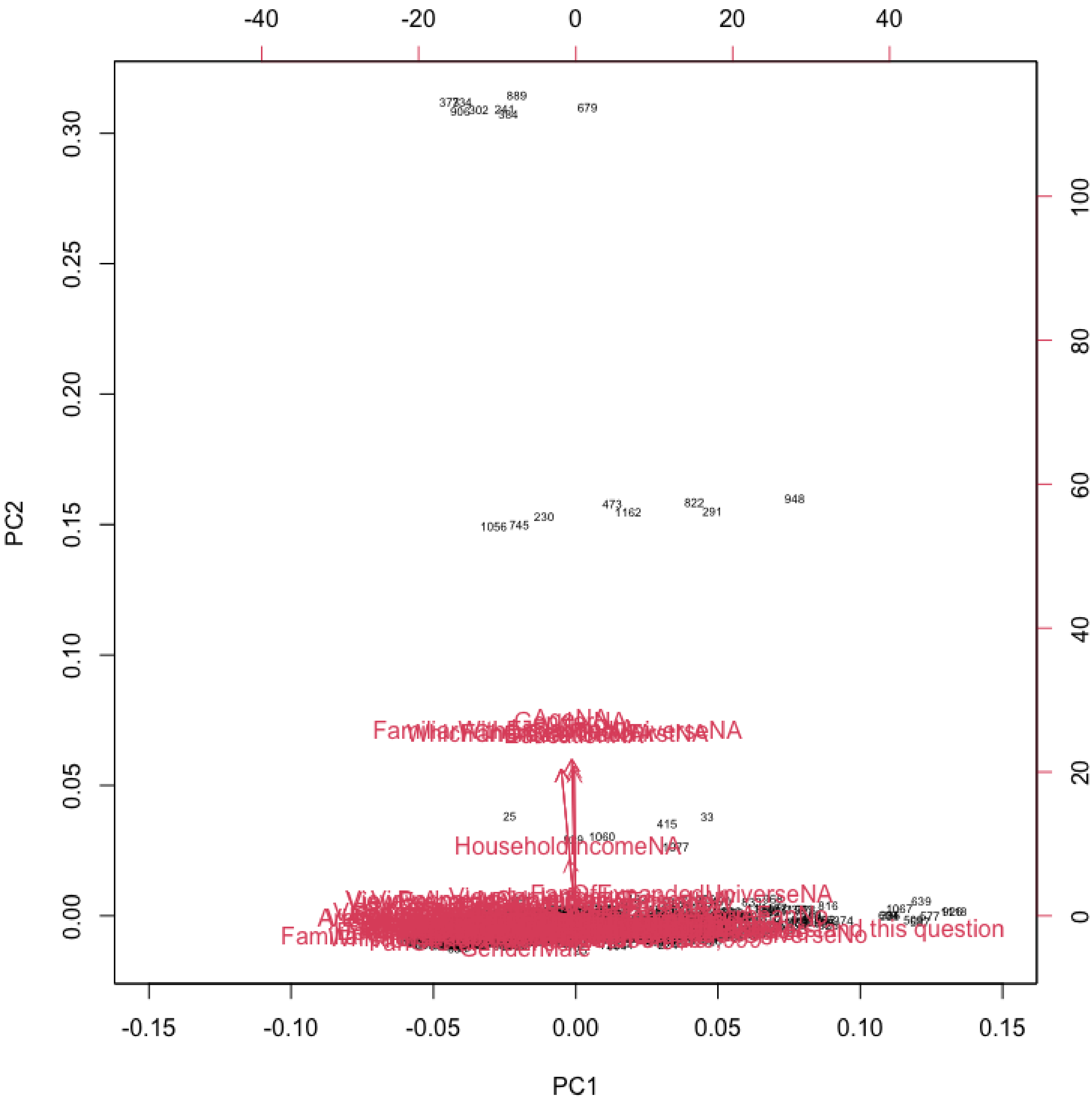
Component1 appears to picks off people who are not fans or have never seen the movies or otherwise provided trash data (lots of missing values, contradictions, etc). Lets subset the data according to this and rerun the analysis, computing principal components on this subset of data.

```
> subset = starwars.x[pca$x[,1]<4,]
> # when we subset the data, we see the same problem we had before!
> # We now have some columns that are totally constant.
> subset = subset[,apply(subset, 2, sd)>0 ]
> pca = prcomp(subset, scale=T)
> plot(jitter(pca$x[,1:2],0.1))
```



Let's take a look at the biplot to see if we can find anything to note about the survey responses. What we notice are the arrows pointing away from the main cloud of data with "NA" tags at the end of them... more missing values. Annoying.

```
> biplot(pca,cex=c(0.5,1), xlim=c(-0.15,0.15))
```



The problem with correlation PCA. (Sometimes.)

It ***still*** looks like this PCA solution will be plagued by poor data quality. Why? Probably because we are forcing the variance of variables to unity using correlation PCA. Let's take a look at the covariance alternative and see if we can find some other trends of information in the data. This will give higher weight to variables with larger scales, namely the likert scale questions in columns 10 through 29, but that should be ok with us given the value (in terms of richness of information) those questions bring to the survey overall.

The Covariance PCA

We'll stick with the subset of higher quality data that we found from our first correlation PCA, but explore what's left with covariance PCA:

```
> pca = prcomp(subset, scale=F)
> plot(pca$x[,1:2])
```

We note that the projection of the data is much more uniform in nature, not broken apart by clusters driven by missing data. Let's go further and see if the biplot is revealing anything interesting.

```
> biplot(pca)
```